TEMA 1. Sistemas Combinacionales.

4. ESTRUCTURAS COMBINACIONALES BÁSICAS (I)

- Puertas lógicas básicas*.
- Multiplexores y demultiplexores.
- Codificadores y decodificadores.
- Comparadores.

* ver transparencias 42 y 43

4.2 MULTIPLEXORES (I)

Un multiplexor de información o selector de datos, es un circuito lógico combinacional que acepta varias entradas de datos y selecciona ("transmite") solamente una de ellas hasta su salida, dependiendo del valor de una señal de control.

Coloca en la salida la misma señal que haya en la entrada QUE SELECCIONAN **las señales de control**

ENTRADA DE DATOS. Nº de entradas: n

ENTRADAS DE CONTROL. Nº de entradas: p $\rightarrow 2^p \geq n$

$2^p = n$ **(en el caso más simple)**

SALIDA (O SALIDAS). Nº de salidas: 1

4.2 MULTIPLEXORES (II)

EJEMPLO para 2 entradas

Bloque funcional

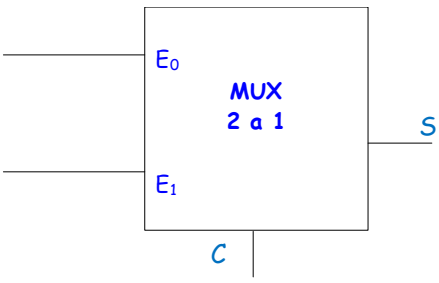


Tabla de verdad

C	S
0	E_0
1	E_1

Función Lógica

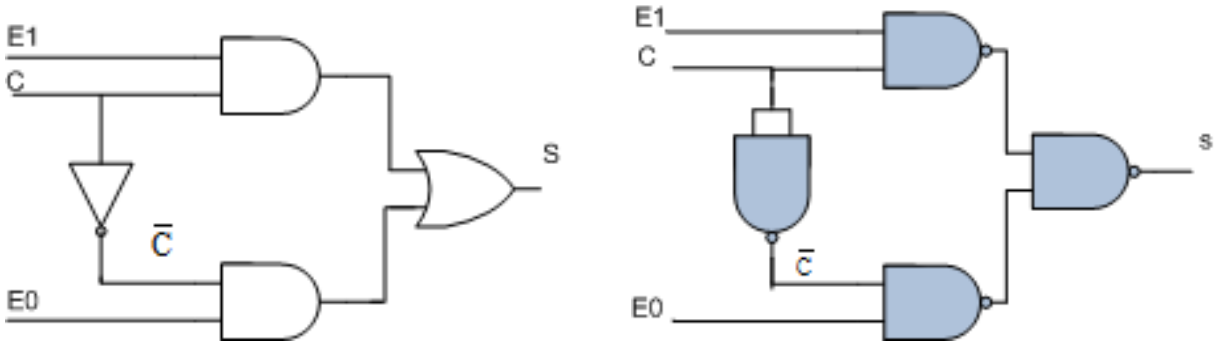
$S\ (MUX\_2{:}1) = E_0\, C' + E_1\, C$

4.2 MULTIPLEXORES (III)

EJEMPLO para 2 entradas

Implementación con puertas lógicas

$$S = E_0\, C' + E_1\, C$$



4.2 MULTIPLEXORES (IV)

EJEMPLO para 4 entradas

Bloque funcional

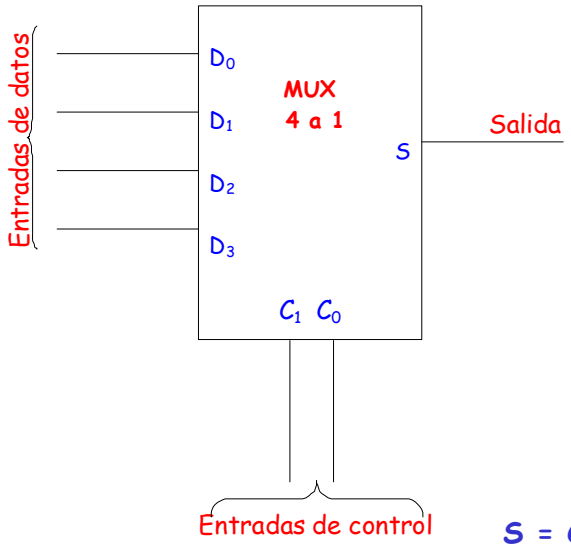


Tabla de verdad

C_1	C_0	S
0	0	D_0
0	1	D_1
1	0	D_2
1	1	D_3

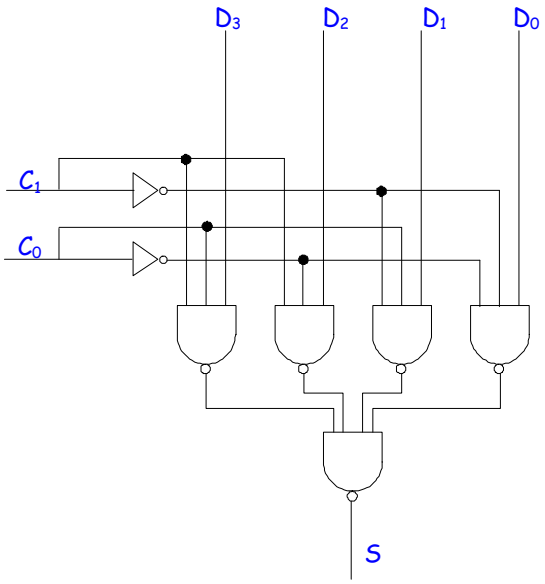
Función Lógica

$$S = C_1'C_0'\,(D_0) + C_1'C_0\,(D_1) + C_1C_0'\,(D_2) + C_1C_0\,(D_3)$$

4.2 MULTIPLEXORES (V)

EJEMPLO para 4 entradas

Implementación con puertas lógicas (NAND)



Función lógica

$$S = D_0\overline{c_1}\,\overline{c_0} + D_1\overline{c_1}c_0 + D_2c_1\overline{c_0} + D_3c_1c_0$$

$$S = \overline{D_0\overline{c_1}\,\overline{c_0} + D_1\overline{c_1}c_0 + D_2c_1\overline{c_0} + D_3c_1c_0}$$

$$S = \overline{D_0\overline{c_1}\,\overline{c_0}}\ \overline{D_1\overline{c_1}c_0}\ \overline{D_2c_1\overline{c_0}}\ \overline{D_3c_1c_0}$$

4.2 MULTIPLEXORES (XI)

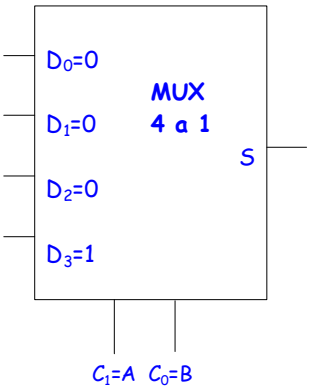
Aplicaciones: implementación de funciones lógicas

Implementación de la función S = A.B (AND) con un MUX 4:1

$S_{(MUX\ 4:1)} = D_0\,(C_1'\,C_0') + D_1\,(C_1'\,C_0) + D_2\,(C_1\,C_0') + D_3\,(C_1\,C_0)$

$S_{(MUX\ 4:1)} = A.B = (D_0=0)(C_1'\,C_0') + (D_1=0)(C_1'\,C_0) + (D_2=0)(C_1\,C_0') + (D_3=1)(C_1\,C_0) =$
$(D_0=0)(A'\,B') + (D_1=0)(A'\,B) + (D_2=0)(A\,B') + (D_3=1)(A\,B) =$

$C_1(A)$	$C_0(B)$	S
0	0	**0**
0	1	**0**
1	0	**0**
1	1	**1**



4.2 MULTIPLEXORES (XII)

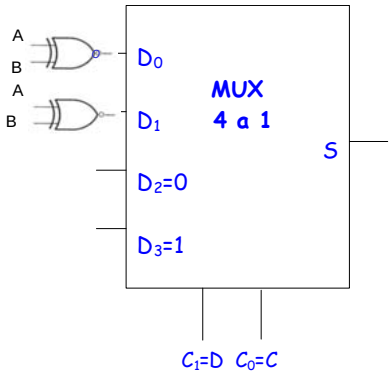
Aplicaciones: implementación de funciones lógicas

S = (B´ A´ + BA) D'C' +(B´ A + BA´)D'C + DC directamente con un MUX 4:1

Ecuación general: ⟹ $S_{(MUX\ 4:1)} = E_0 (C_1´ C_0´) + E_1 (C_1´ C_0) + E_2 (C_1 C_0´) + E_3 (C_1 C_0)$

$S_{(MUX\ 4:1)} = (D_0 \gg (A \varnothing B)´)(D´ C´) + (D_1 \gg A \varnothing B)(D´ C) + (D_2 \gg 0)(DC´) + (D_3 \gg 1)(D\ C)$

C_1(D)	C_0(C)	S
0	0	(AØB)´
0	1	(AØB)
1	0	0
1	1	1



4.2 MULTIPLEXORES (XIII)

Aplicaciones: implementación de funciones lógicas

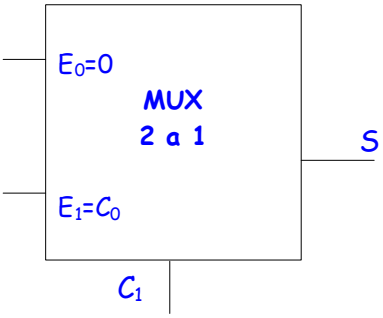
Implementación introduciendo en las entradas del MUX alguna de las variables de la función.

EJEMPLO: S = A.B (AND)

C_1(A)	C_0(B)	S
0	0	0
0	1	0
1	0	0
1	1	1

S = 0 cuando C_1 = 0
S = C_0 cuando C_1 = 1

S = 0 cuando A=0
S= B cuando A=1

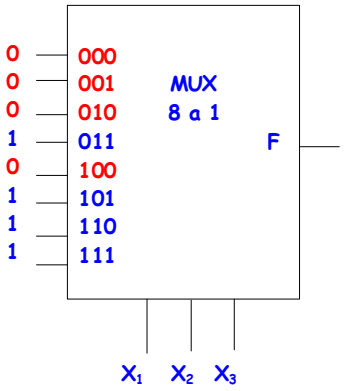


$S_{(MUX\ 2:1)} = (E_0 = 0)\ C_1´ + (E_1 = C_0)\ C_1 = (E_0 = 0)\ A´ + (E_1 = B)\ A = AB$

4.2 MULTIPLEXORES (XIV)

Sea la función F ($x_1x_2x_3$) dada por la tabla de verdad adjunta, se puede construir utilizando un multiplexor con entradas valores constantes (0, 1) y tomando como variables de control ($x_1x_2x_3$)

x_1	x_2	x_3	F
0	0	0	0
0	0	1	0
0	1	0	0
0	1	1	1
1	0	0	0
1	0	1	1
1	1	0	1
1	1	1	1



4.2 MULTIPLEXORES (XV)

La misma función F($x_1x_2x_3$), se puede construir con entradas que dependan de una variable (x_3) y tomando como variables de control (x_1 x_2):

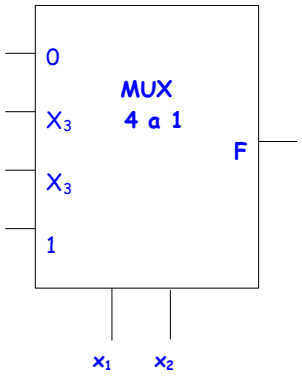
x_1	x_2	x_3	F
0	0	0	0
0	0	1	0
0	1	0	0
0	1	1	1
1	0	0	0
1	0	1	1
1	1	0	1
1	1	1	1

F = 0 cuando $x_1 = x_2 = 0$

F = X_3 cuando $x_1 = 0$ y $x_2 = 1$

F = X_3 cuando $x_1 = 1$ y $x_2 = 0$

F= 1 cuando $x_1 = x_2 = 1$



4.2 MULTIPLEXORES (XVI)

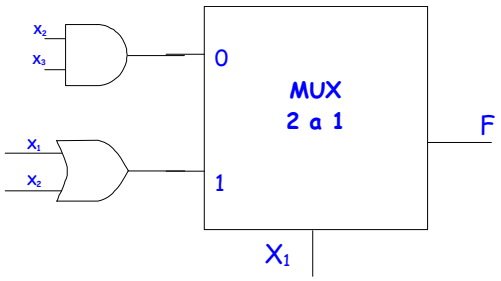
La misma función $F(x_1x_2x_3)$, se puede construir con entradas que dependan de las 2 variables (x_2 x_3) y tomando como variable de control (x_1):

x_1	x_2	x_3	F
0	0	0	0
0	0	1	0
0	1	0	0
0	1	1	1
1	0	0	0
1	0	1	1
1	1	0	1
1	1	1	1

F será función de x_2 y de x_3

$F = x_2 x_3$ cuando $x_1 = 0$

$F = x_1 + x_2$ cuando $x_1 = 1$



4.2 MULTIPLEXORES (XVII):

TEOREMA DE EXPANSIÓN DE SHANNON

- $F(A_1, A_2, \ldots, A_N) = \overline{A_1} \cdot F(0, A_2, \ldots, A_N) + A_1 \cdot F(1, A_2, \ldots, A_N)$

- $F(A_1, A_2, \ldots, A_N) = (\overline{A_1} + F(0, A_2, \ldots, A_N)) \cdot (A_1 + F(1, A_2, \ldots, A_N))$

Si se emplea repetidamente el teorema de expansión de *Shannon*, hasta agotar todas las variables, nos proporciona directamente el desarrollo de una función como:

SUMA DE TÉRMINOS PRODUCTO (*MINTERMS*) o como

PRODUCTO DE TÉRMINOS SUMA (*MAXTERMS*).

4.2 MULTIPLEXORES (XVIII):

TEOREMA DE EXPANSIÓN DE SHANNON

Sabemos que:

$$F(A_1,A_2) = \overline{A_1}\cdot F(0,A_2) + A_1\cdot F(1,A_2) =$$
$$= \overline{A_1}\cdot\overline{A_2}\cdot F(0,0) + \overline{A_1}\cdot A_2\cdot F(0,1) + A_1\cdot\overline{A_2}\cdot F(1,0) + A_1\cdot A_2\cdot F(1,1)$$

Hay que tener en cuenta que:

- $\overline{A_1}\cdot F(0,A_2) = \overline{A_1}\cdot\overline{A_2}\cdot F(0,0) + \overline{A_1}\cdot A_2\cdot F(0,1)$
- $A_1\cdot F(1,A_2) = A_1\cdot\overline{A_2}\cdot F(1,0) + A_1\cdot A_2\cdot F(1,1)$

- F(0,0), F(0,1), F(1,0) y F(1,1) solamente podrán tomar los valores: '1' o '0'.

4.2 MULTIPLEXORES (XIX):

TEOREMA DE EXPANSIÓN DE SHANNON

Ejemplo:

Si para el caso anterior tenemos la siguiente tabla de verdad...

X_1	X_2	$F(X_1, X_2)$
0	0	0
0	1	1
1	0	1
1	1	0

$$F(X_1,X_2)=\overline{X_1}\cdot\overline{X_2}\cdot F(0,0)+\overline{X_1}\cdot X_2\cdot F(0,1)+X_1\cdot\overline{X_2}\cdot F(1,0)+X_1\cdot X_2\cdot F(1,1)=$$
$$= \overline{X_1}\cdot\overline{X_2}\cdot(0)+ \overline{X_1}\cdot X_2\cdot(1)+ X_1\cdot\overline{X_2}\cdot(1)+ X_1\cdot X_2\cdot 0= \overline{X_1}\cdot X_2+ X_1\cdot\overline{X_2}$$

4.2 MULTIPLEXORES (XX)

- **Generación de funciones lógicas**
  - Un multiplexor con N entradas de control permite generar cualquier función lógica de N+1 variables.
  - Siguiendo el teorema de Shannon:

$$F(X_1, X_2, \ldots, X_N, X_M) = \overline{X}_1 F(0, X_2, \ldots, X_M) + X_1 F(1, X_2, \ldots, X_M) =$$
$$= \overline{X}_1\overline{X}_2 \ldots \overline{X}_N F(0, 0, \ldots, 0, X_M) + X_1\overline{X}_2 \ldots \overline{X}_N F(1, 0, \ldots, 0, X_M) + \ldots +$$
$$+ X_1X_2 \ldots X_N F(1, 1, \ldots, 1, X_M) =$$
$$= \overline{X}_1\overline{X}_2 \ldots \overline{X}_N G_0(X_M) + X_1\overline{X}_2 \ldots \overline{X}_N G_1(X_M) + \ldots + X_1X_2 \ldots X_N G_{2N-1}(X_M)$$

  - Las funciones $G_i(X_M)$ serán del tipo: $\{1, 0, X_M, \overline{X}_M\}$

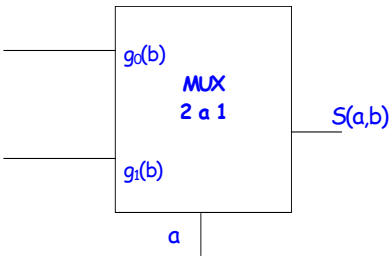
$F(a,b,c) = ab + \overline{b}c$

4.2 MULTIPLEXORES (XXI):

GENERADOR UNIVERSAL: EJEMPLO DE IMPLEMENTACIÓN PARA FUNCIONES DE DOS VARIABLES

Función S (a,b)	Puerta Lógica	Entradas g_0	Entradas g_1
ab	AND	0	b
a + b	OR	b	1
a´b´	NOR	b´	0
a´ + b´	NAND	1	b´
ab´ + a´b	X-OR	b	b´
ab + a´b´	X-NOR	b´	b

$$S(ab) = \bar{a}\, S(0\ b) + a\, S(1\ b)$$
$$= \bar{a}\, g_0 + a\, g_1$$

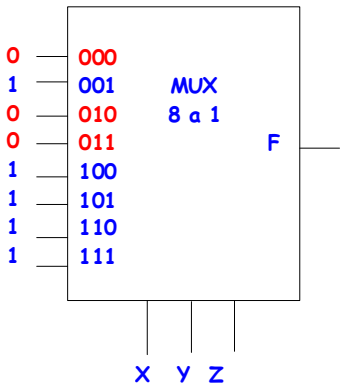


4.2 MULTIPLEXORES (XXII):

Sea la función: $F = X + \overline{Y}Z$,

1. Construimos la tabla de verdad que representa a esta función
2. Construimos una matriz que la implementa, tomando X, Y y Z como variables de control

X	Y	Z	F (X Y Z)
0	0	0	0
0	0	1	1
0	1	0	0
0	1	1	0
1	0	0	1
1	0	1	1
1	1	0	1
1	1	1	1



4.2 MULTIPLEXORES (XXIII):

Sea la función: $F = X + \overline{Y}Z$

3. Desarrollamos por el teorema de Shannon en la variable Z; las entradas al multiplexor dependerán de la variable Z y se toman como variables de control X e Y

$$F(X\,Y\,Z) = \overline{X}\,\overline{Y}\,F(0\,0\,Z) + \overline{X}\,Y\,F(0\,1\,Z) + X\,\overline{Y}\,F(1\,0\,Z) + X\,Y\,F(1\,1\,Z) =$$

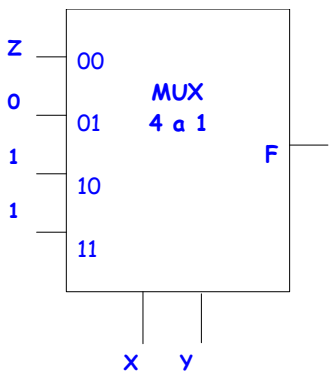
$$= \overline{X}\,\overline{Y}\,(Z) + \overline{X}\,Y\,(0) + X\,\overline{Y}\,(1) + X\,Y\,(1)$$

Siendo F(00Z) = z

F(01Z) = 0

F(10Z) = 1

F(11Z) = 1



4.2 MULTIPLEXORES (XXIV):

Ejemplo:

Función F= X+ $\overline{Y}Z$

La negada de dicha función es:

F= $\overline{X}$ (Y+ $\overline{Z}$)

- Cuando 2 puertas estén situadas en serie → producto booleano.
- Cuando 2 puertas estén situadas en paralelo → suma booleana.
- Casos con salida a "1" separados de casos con salida a "0".

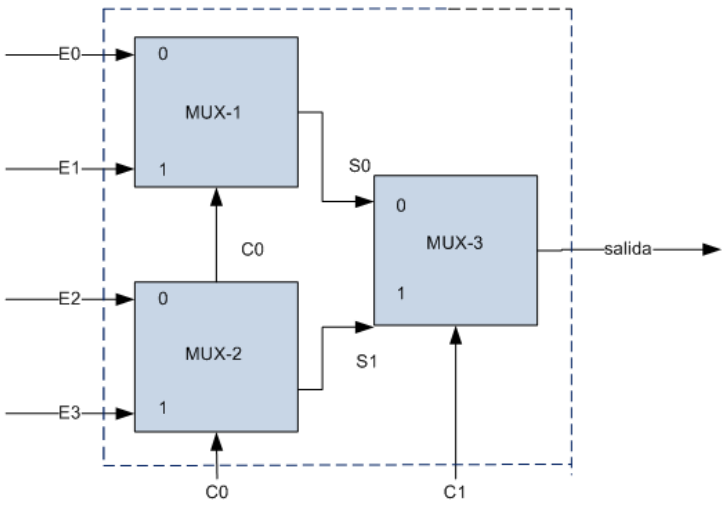
Restricciones

1. Un nodo no puede estar sometido simultáneamente a 2 tensiones diferentes. (no puede haber conducción por dos o más líneas simultáneamente)
2. No puede haber ninguna combinación de las entradas para cual la salida no esté definida. (siempre ha de haber conducción al menos por una línea)

4.2 MULTIPLEXORES (XXV)

EXPANSIÓN DE MULTIPLEXORES: construcción de multiplexores de órdenes superiores, utilizando multiplexores de órdenes inferiores.

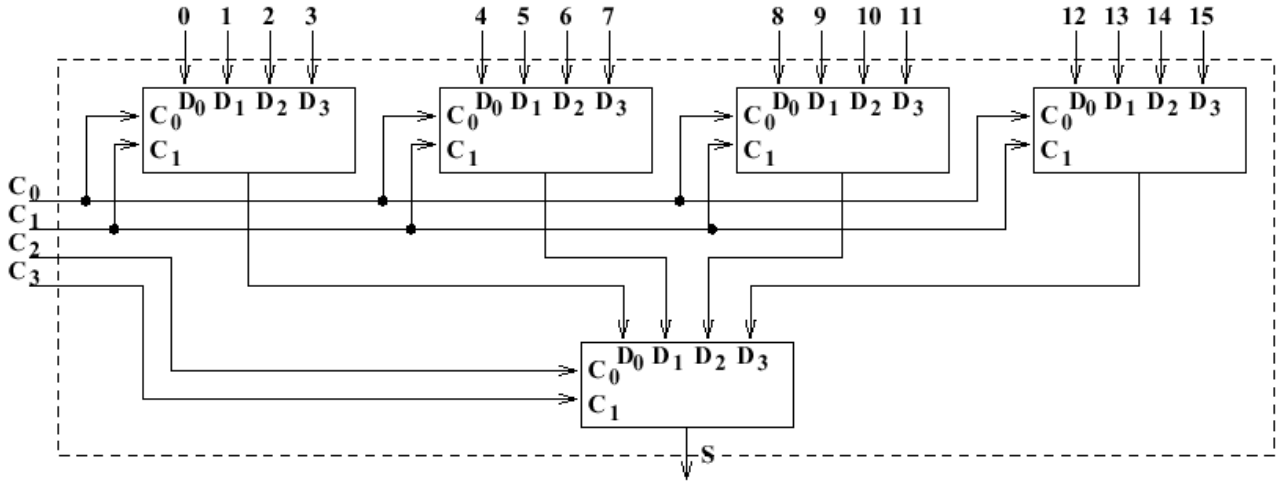
EJEMPLO: MUX 4:1 con multiplexores 2:1



Señales control		Salidas intermedias y general		
C_1	C_0	S_0	S_1	Salida
0	0	E_0	E_2	E_0
0	1	E_1	E_3	E_1
1	0	E_0	E_2	E_2
1	1	E_1	E_3	E_3

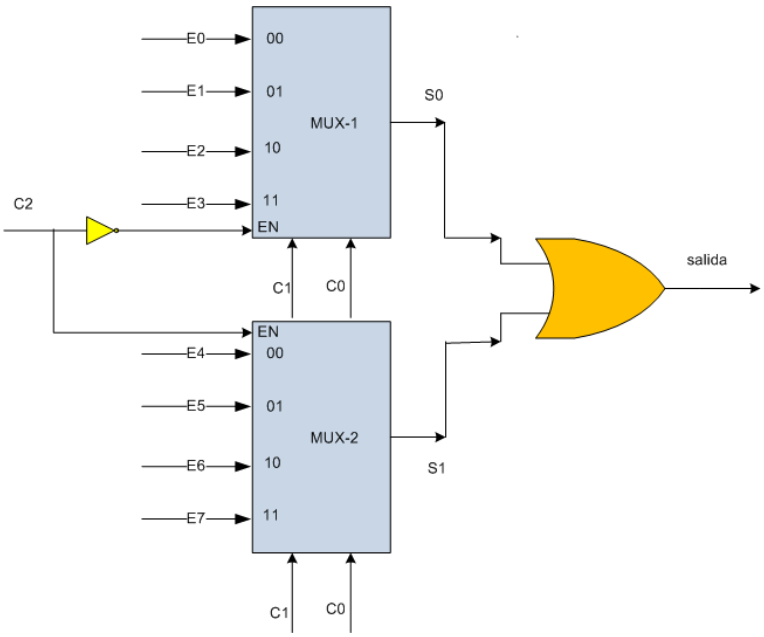
4.2 MULTIPLEXORES (XXVI)

EJEMPLO: MUX 16:1 con multiplexores 4:1



4.2 MULTIPLEXORES (XXVII)

EJEMPLO: MUX 8:1 con señal ENABLE y construido con multiplexores 4:1



C2	C1	C0	S0	S1	Salida
0	0	0	E0	0	E0
0	0	1	E1	0	E1
0	1	0	E2	0	E2
0	1	1	E3	0	E3
1	0	0	0	E4	E4
1	0	1	0	E5	E5
1	1	0	0	E6	E6
1	1	1	0	E7	E7

4.2 DEMULTIPLEXORES (I)

Un demultiplexor realiza la operación inversa de la efectuada por un multiplexor.

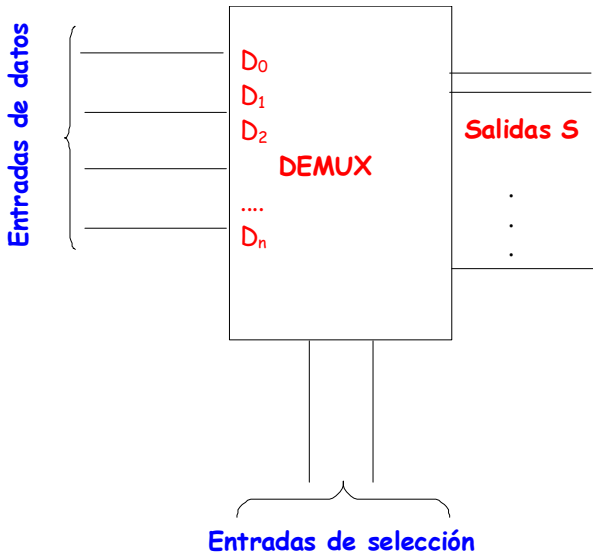
Distribuye entre las salidas la señal que haya tomado de una entrada seleccionada por las señales de control

ENTRADA DE DATOS. Nº de entradas: m
ENTRADAS DE SELECCIÓN. Nº de entradas: p $\rightarrow 2^p \leq n$
SALIDA (O SALIDAS). Nº de salidas: n

Puede verse como un decodificador en el que la entrada de activación o *enable* (E) se utiliza como entrada de datos.

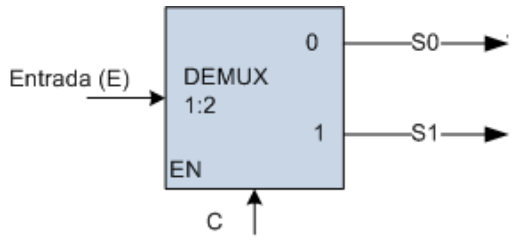
4.2 DEMULTIPLEXORES (II)

Bloque funcional



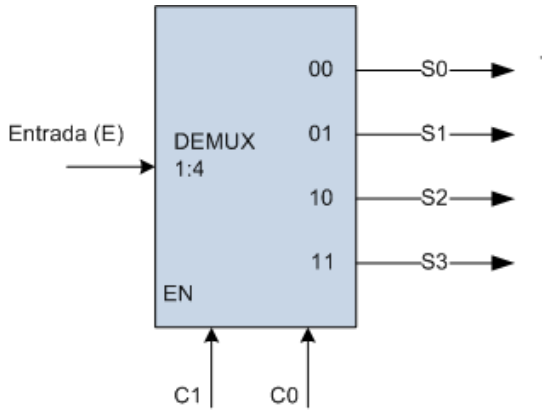
4.2 DEMULTIPLEXORES (III)

EJEMPLOS: DEMUX 1:2 y DEMUX 1:4



Control	Salidas	
C	S1	S0
0	0	E
1	E	0

S0 = EC'
S1 = EC



Control		Salidas			
C_1	C_0	S3	S2	S1	S0
0	0	0	0	0	E
0	1	0	0	E	0
1	0	0	E	0	0
1	1	E	0	0	0

S0 = E C1' C0'
S1 = E C1´C0
S2 = E C1 C0'
S3 = E C1 C0

4.2 DEMULTIPLEXORES (IV)

Demultiplexor 1:8, tabla de verdad

S_2	S_1	S_0	O_7	O_6	O_5	O_4	O_3	O_2	O_1	O_0
0	0	0	0	0	0	0	0	0	0	I
0	0	1	0	0	0	0	0	0	I	0
0	1	0	0	0	0	0	0	I	0	0
0	1	1	0	0	0	0	I	0	0	0
1	0	0	0	0	0	I	0	0	0	0
1	0	1	0	0	I	0	0	0	0	0
1	1	0	0	I	0	0	0	0	0	0
1	1	1	I	0	0	0	0	0	0	0

$(O_7\ O_6\ O_5\ O_4\ O_3\ O_2\ O_1\ O_0) => F\ (S_2\ S_1\ S_0)$

EJERCICIO DE APLICACIÓN: constrúyase con puertas AND

4.3 CODIFICADORES Y DECODIFICADORES (I)

Un Codificador binario es un circuito lógico combinacional que para cada entrada activada, produce un código de salida (combinación lógica) de N bits.

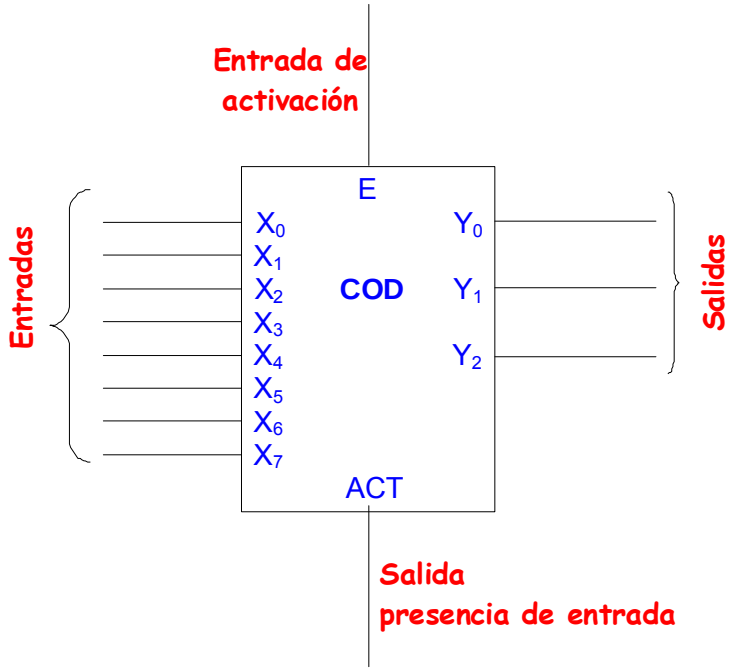
Para cada entrada activa un código de salida

ENTRADA DE DATOS. Nº de entradas: m
SALIDAS Nº de salidas: n
En el caso general: n y m $\in$ Z y se cumple $2^n \geq m$

en el caso particular: $m = 2^n$

4.3 CODIFICADORES (II)

Bloque funcional



4.3 CODIFICADORES (III)

Codificador de octal a binario. Tabla de verdad

	A_7	A_6	A_5	A_4	A_3	A_2	A_1	A_0		O_2	O_1	O_0
	0	0	0	0	0	0	0	1		0	0	0
	0	0	0	0	0	0	1	0		0	0	1
	0	0	0	0	0	1	0	0		0	1	0
	0	0	0	0	1	0	0	0		0	1	1
	0	0	0	1	0	0	0	0		1	0	0
	0	0	1	0	0	0	0	0		1	0	1
	0	1	0	0	0	0	0	0		1	1	0
	1	0	0	0	0	0	0	0		1	1	1

EJERCICIO DE APLICACIÓN: Impleméntese con puertas OR

4.3 DECODIFICADORES (I)

Un decodificador es un circuito lógico combinacional que para cada combinación lógica de los valores de sus entradas activa una y solamente una salida.

Para cada código de entrada se activa una salida

ENTRADA DE DATOS. Nº de entradas: n
SALIDAS Nº de salidas: m

n y m $\in Z$ y cumplen $m \leq 2^n$

Las salidas de un decodificador generan todos los productos canónicos de las entradas

4.3 DECODIFICADORES (II)

Bloque funcional

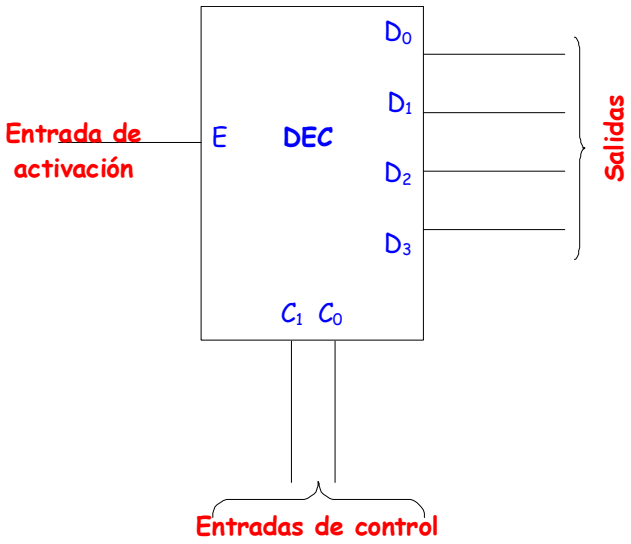
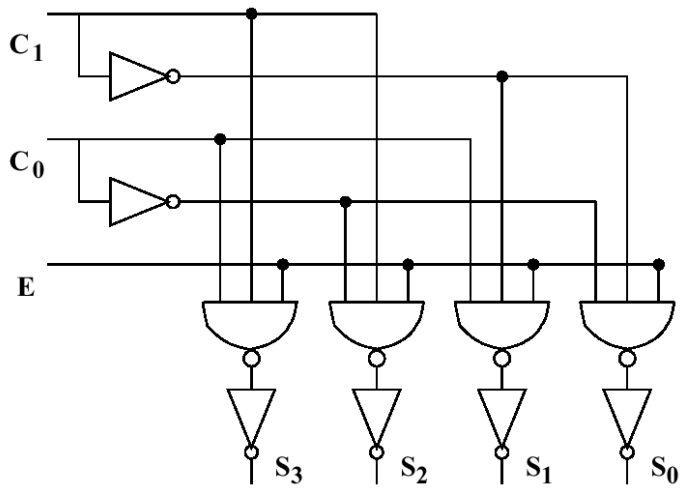


Tabla de verdad

E	C_1	C_0	D_3	D_2	D_1	D_0
0	x	x	0	0	0	0
1	0	0	0	0	0	1
1	0	1	0	0	1	0
1	1	0	0	1	0	0
1	1	1	1	0	0	0

4.3 DECODIFICADORES (III)

Estructura interna basada en puertas lógicas NAND



Funciones lógicas de salida

$S_0 = E\overline{C_1}\,\overline{C_0}$

$S_1 = E\overline{C_1}C_0$

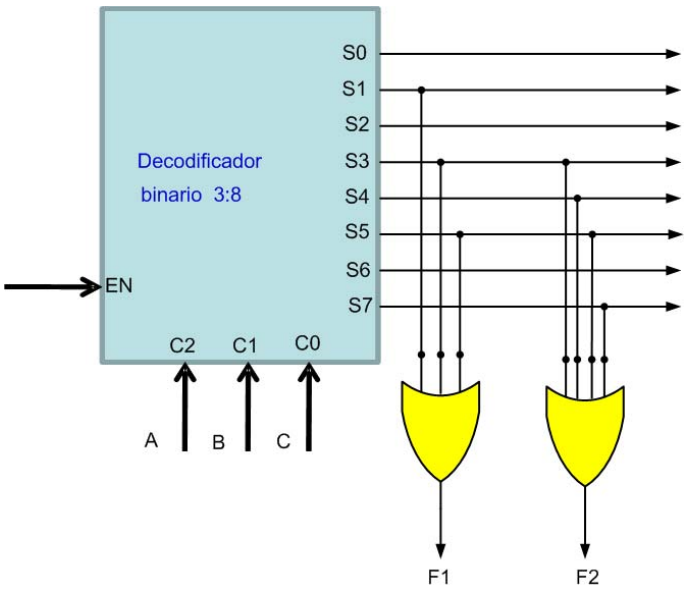
$S_2 = EC_1\overline{C_0}$

$S_3 = EC_1C_0$

Ejemplo: $F(a,b,c) = ab + \overline{bc}$

4.3 DECODIFICADORES (IV)

Ejemplo de utilización de decodificadores para la implementación de funciones lógicas



Sean las funciones de 3 variables

F1 = B'C + A'C y

F2 = AB' + BC

Transformación de las funciones a sus formas canónicas:

F1 = B'C + A'C =

= A'B'C+ AB'C + A'B'C+ A'BC =

= Σ (1, 3, 5)

F2 = AB' + BC =

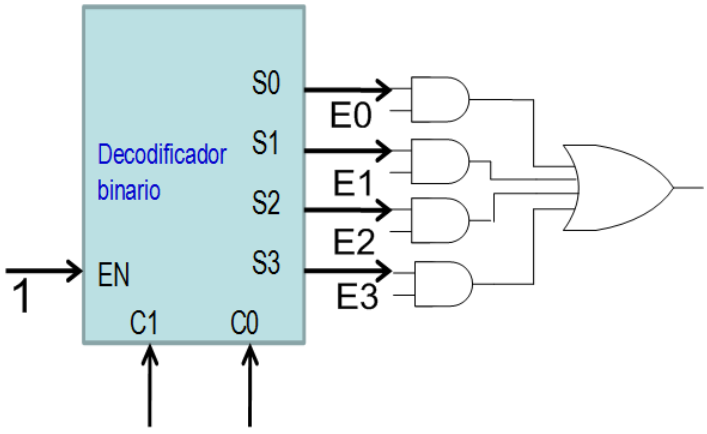
= AB'C' + AB'C + A'BC + ABC =

= Σ (3, 4, 5, 7)

Al tener 3 variables se requiere un decoder de 3:8 (3 variables de control)

4.3 DECODIFICADORES (V)

EJEMPLO: Construcción de un multiplexor 4:1, con un decodificador 2:4 y las puertas lógicas que se precisen



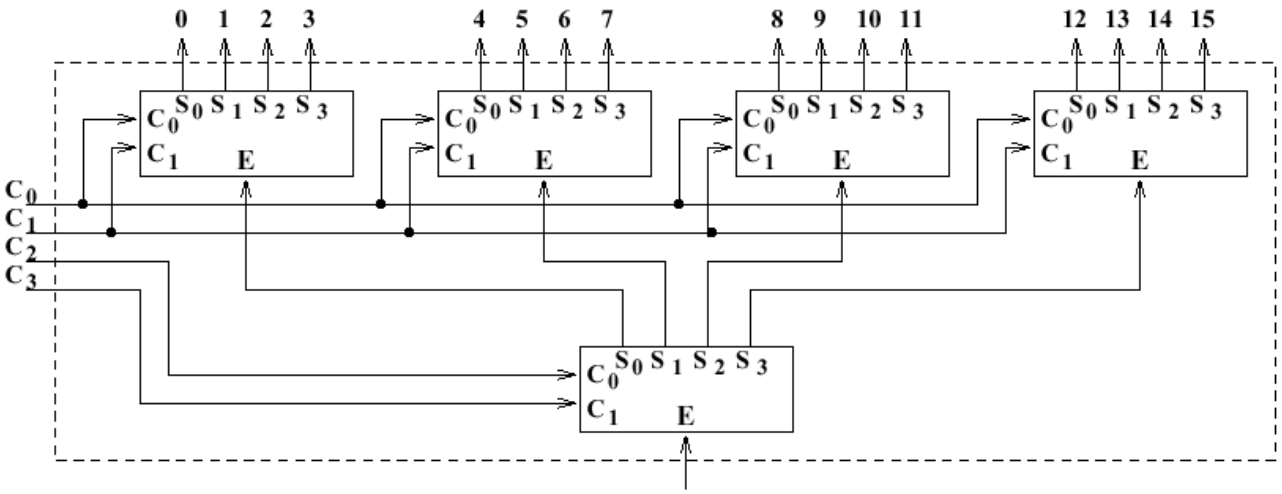
EN	C_1	C_0	S_3	S_2	S_1	S_0
0	x	x	0	0	0	0
1	0	0	0	0	0	1
1	0	1	0	0	1	0
1	1	0	0	1	0	0
1	1	1	1	0	0	0

Control		MUX
C1	C0	Salida
0	0	E_0
0	1	E_1
1	0	E_2
1	1	E_3

4.3 DECODIFICADORES (VI)

Obtención de decodificadores de órdenes superiores

EJEMPLO: decodificador 4:16 basado en decodificadores 2:4

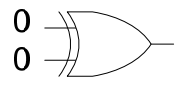


4.4 COMPARADORES (I)

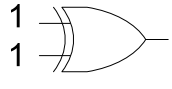
Los COMPARADORES son sistemas combinacionales que comparan las magnitudes de cantidades binarias para determinar su relación.

Determinación de IGUALDAD:

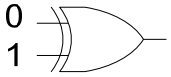
Comparación de números de 1 bit



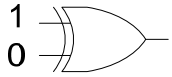
0 Los bits de entrada son IGUALES



0 Los bits de entrada son IGUALES



1 Los bits de entrada son DISTINTOS



1 Los bits de entrada son DISTINTOS

4.4 COMPARADORES (II)

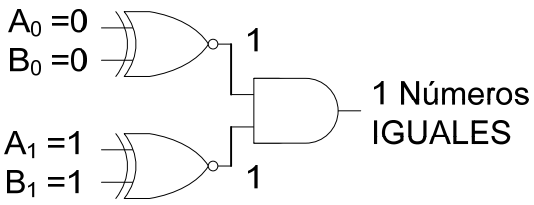
Comparación de números de 2 bits

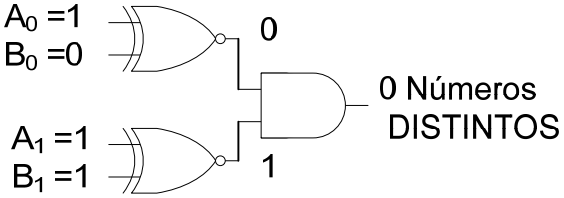
A_1A_0
B_1B_0

Menor peso: subíndice 0

10
10

11
10





A B	XNOR
00	1
01	0
10	0
11	1

4.4 COMPARADORES (III)

Comparación de números de 4 bits

Menor peso: subíndice 0

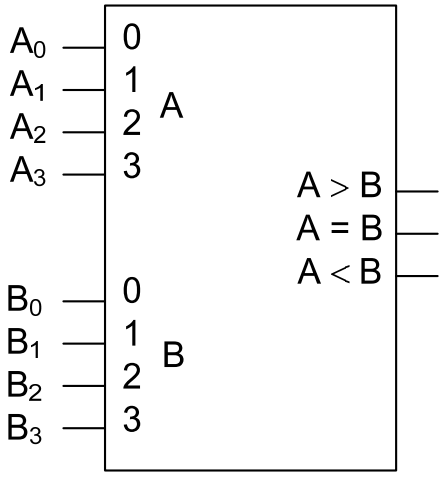
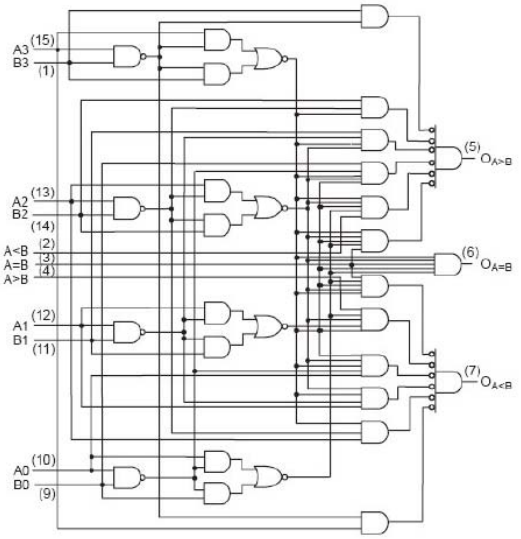


Diagrama lógico del 74LS85



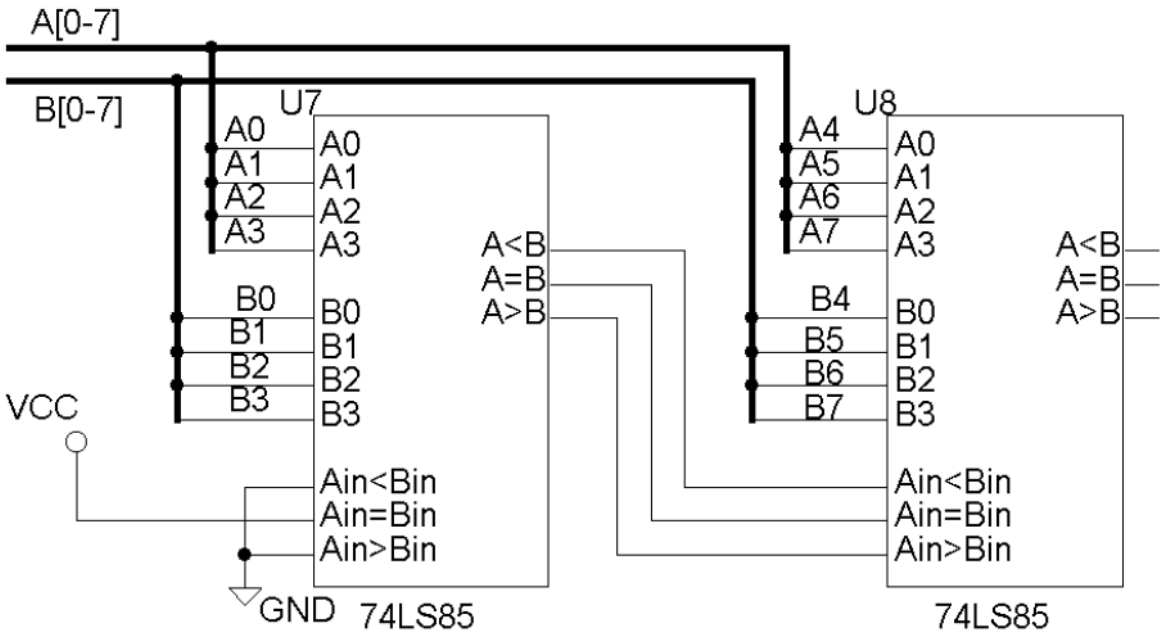
Si A_3=1 y B_3=0 → A > B

Si A_3=0 y B_3=1 → A < B

Si A_3= B_3 → Hay que examinar los otros bits

4.4 COMPARADORES (IV)

Comparación de números de 8 bits



4.4 COMPARADORES (V)

Comparación de números de 16 bits